Test System for ABCD3T Wafer Screening

Prepared by: A. Ciocio and V. Fadeyev
Lawrence Berkeley National Laboratory, Berkeley, CA

June 20, 2001

Table of Contents

1. Introduction

We describe the ASIC wafer test system that has been developed to provide comprehensive production screening of the ATLAS Semiconductor Tracker front-end chip (ABCD3T). The ABCD3T features a 128-channel analog front-end, a digital pipeline, and communication circuitry, clocked at 40 MHz, which is the bunch crossing frequency at the LHC. The tester measures values and tolerance ranges of all critical IC parameters, including DC parameters, electronic noise, time resolution, I/O signal levels and clock timing. The tester is controlled by an FPGA (ORCA3T) programmed to issue the input commands to the IC and to interpret the output data. This allows the high-speed wafer-level IC testing necessary to meet the production schedule. To characterize signal amplitudes and phase margins, the tester utilizes pin-driver, delay, and DAC chips, which control the amplitudes and delays of signals sent to the IC. Output signals from the IC go through window comparator chips to measure their levels. A probe card has been designed specifically to reduce pick-up noise that can affect the measurements. The system can operate at frequencies up to 100 MHz to study the speed limits of the digital circuitry before and after radiation damage.
In Section 2 we describe the system requirements for the IC's analogue and digital tests, the system layout and dataflow. The description of the analogue and digital tests is presented in Section 3 and the error handling in Section 4. The implementation is described in Section 5.

2. System Description

2.1. Testing Requirements

The ABCD3T[1], which operates at 40 MHz, features a 128-channel analog front-end consisting of amplifiers and comparators, and a digital pipeline and communication circuitry. To reduce power consumption and cost, the IC utilizes a binary readout scheme where the signals from the silicon detector are amplified and then compared to a threshold. Only the result of this comparison based on a hit or no-hit logic is stored in the digital pipeline. Internal 4-bit DACs to trim the threshold are part of each readout channel to correct for the gain and threshold variations both before and after radiation damage. The analog circuitry of the chip is calibrated via a charge injection method.
The basic method for the characterization of the analog circuitry of the ABCD3T consists of an efficiency scan for different threshold values with a fixed calibration charge. A typical "S-curve" from this measurement is shown in Fig. 1. The 50% point indicates the threshold voltage corresponding to the injected charge and the width characterizes the noise. To extract the gain and offsets, the scan is done at different calibration input charges. To characterize the trim DACs, the scan must be done for all the different DAC values. All of these procedures can be performed automatically for each of the 128 channels (Figures 2,3) on every IC on the wafer. To avoid long testing times, the histograms of the data from the chip are calculated in hardware.

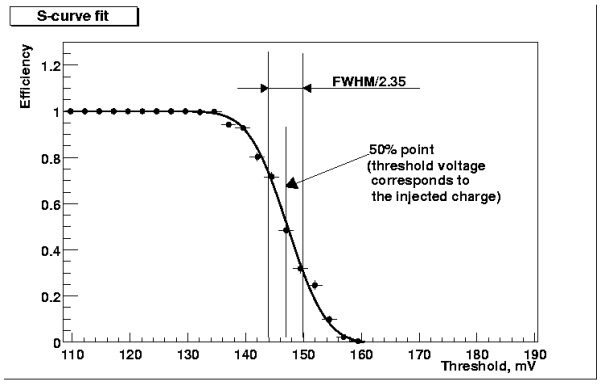

Figure 1. *S-curve for a single channel.*

The digital part of the chip includes redundancy and data bypassing circuitry, to mitigate the effect of circuit failure in the high radiation environment of the LHC. The functions of the digital circuitry of the IC are verified using test vectors, which define the sequence of the IC's control line values for consecutive clock cycles. The IC's output signals are received by the FPGA and compared with

the expected data corresponding to that particular test vector. Only the result of the comparison is read out. It has been shown that the radiation damage slows some digital circuits. Thus the

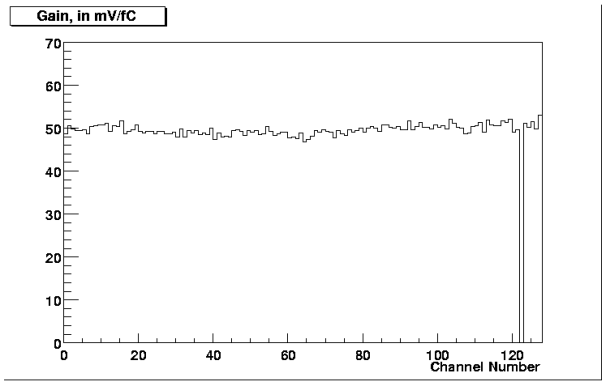

Figure 2. *Gain in mV/fC for 128 channels*

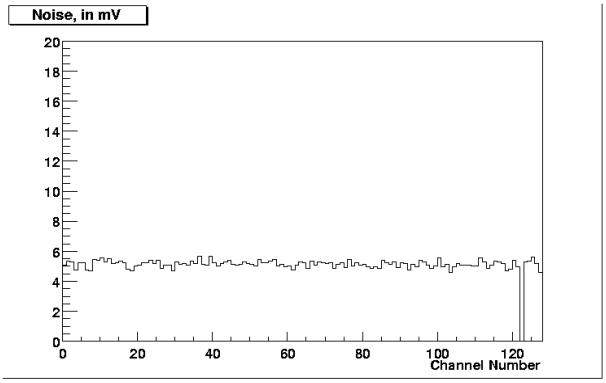

Figure 3. *Noise in mV of 128 channels of one ABCD3T*

ABCD3T should operate at frequency well above 40 MHz before irradiation. To evaluate ABCD3T performance after irradiation, the test vectors are run at frequencies higher than the nominal 40 MHz.

To satisfy the schedule for the construction of the ATLAS SCT, the required wafer screening production rate exceeds one wafer per day per testing site (65536 channels). Therefore a very fast system is required.

2.2. Layout of the Test System.

The ASIC wafer test system[2], which consists of several custom designed PC boards and control software, has been conceived and built to meet all of the ABCD3T testing requirements (Fig. 4).

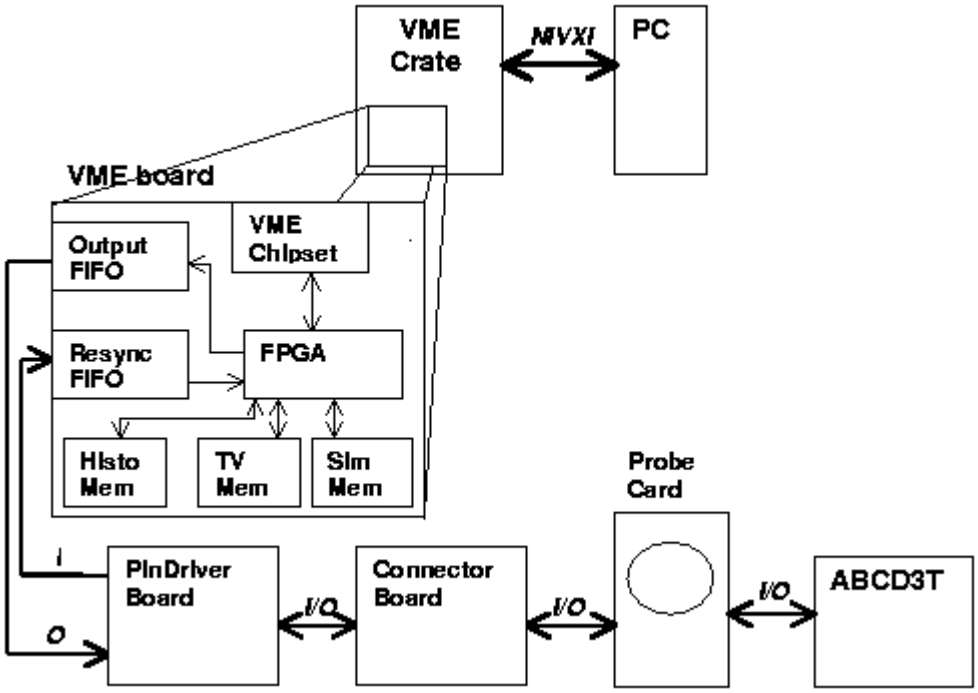

Figure 4. *Architecture of the ASIC test system*

A custom-designed VME board is equipped with an ORCA3T FPGA of 186000 gates, operating at 40 MHz and programmed in VHDL[3].
The main FPGA functions are:

1) To interpret VME commands and generate the bit-streams to be sent to the ABCD3T
2) To interpret the bitstream produced by the ABCD3T during the analog calibration procedure and store the histogram of hits in a dedicated memory chip on the board
3) To store a test vector and the simulated response (simulation vector) of the chip in a memory on the board, to send the test vector to the chip, compare the chip response with the simulation vector and provide the result of the comparison. This procedure is used for digital testing of the ABCD3T.

The test vectors can be run at higher frequencies than the nominal 40 MHz (up to 100 MHz).
This capability is achieved by buffering the signals in FIFOs, which allow for different frequencies of read/write operations. The higher frequency is obtained using a phase lock loop circuit on the VME board. The frequency is set by the FPGA.
The data between the VME board and the probe card are transmitted as differential signals. Two intermediate boards[4] have been designed and built to provide:

1) High current pin drivers for inputs with variable signal level
2) Signal delays
3) Window comparators discriminating on the upper and lower output signal levels
4) Dedicated ADCs to probe the internal ABCD3T parameters; power consumption and feedback control of voltages supplied to the chip and ambient temperature of the pin driver board.

For all signals, the pin-driver and window comparator levels and delays are controlled by the FPGA via dedicated DACs.
A probe card[5] has been designed specifically to reduce the pick-up noise that can affect the analog measurements. Digital and analog signals are separated on different planes. An analog ground layer as well as a split digital/analog ground layer are used and low frequency filters are applied to the differential lines that control the analog part of the chip. All the decoupling capacitors are located as close as possible to the probe pins.

2.3. Data Flow and Controls

All VME operations are done via programmable IO (no block transfers) by simply writing to and reading from VME addresses, one at a time (Appendix A). These read/write commands are passed to the FPGA on the VME board, which is the "brain" of the system. The FPGA controls the data flow, does histogramming, test vector comparison, and controls DAC/ADC operations.
The FPGA firmware consists of several pieces (modules). The simplified diagram of the FPGA partitioning is shown in Figure 5. The arrows indicate the major data flow directions. Control signals are not shown.
The "Histogramming", "TV", "ABCD commands", "Frequency" and "DAC/ADC" modules inside the FPGA are all independent from each other, with the caveat that the first two use the same I/O lines. One can use DAC/ADC functionality while doing other things. All modules have state machines, which start going through a cycle after the corresponding signal from VME. The cycle is always completed. If a second VME command addressing the ***same*** module happens too soon, then it would be ignored. The minimal measured time between consecutive VME bus accesses is 1.8 μs. This corresponds to 72 clock cycles and is enough for most instructions to be completed. There are four notable exceptions, which may take more time:

1) "start sending triggers and histogram the data",
2) "clear histogram memory",
3) "send test vector",
4) "send convert signal to ADC".

To provide feedback about the internal states after the four commands, the BUSY signals were introduced. The user must wait for the corresponding BUSY bit to clear after any of the four commands.

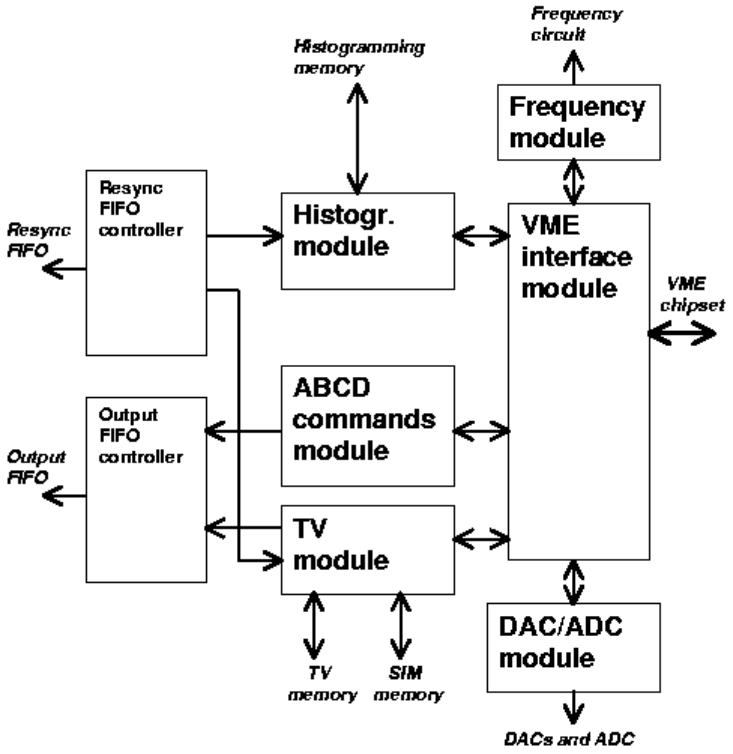

Figure 5. *FPGA modules*

2.3.1 The FPGA Modules

The following is a general description of the functionality of the main FPGA code modules.

VME interface Module

This module decodes the addresses of the VME read/write operations. It is used as an interface to transfer the data and control signals to and from the other modules.
This module has a Status Register which contains the following information:
a) *a fixed pattern on the lowest bits* useful for testing the availability of the VME communication.
b) bits used to define when the histogramming, TV and DAC/ADC modules are busy
c) the result of running a test vector

Frequency Module

Most parts on the VME board, including the FPGA, run at a fixed frequency of 40 MHz. However, the FIFOs can talk to the outside world at higher frequency. In this case the rest of the hardware (pin driver and connector boards and probe card) will be forced to operate at this high frequency. We use this functionality for running test vectors only.
To obtain the high frequency, we use a phase-lock-loop (PLL) with 20 MHz clock (40 MHz divided in half) as an input reference. The frequency module in the FPGA transforms the input parameters needed to program PLL into a serial bitstream and sends it out. The output frequency (in MHz) is

Fout = (Fin/8) * (M/N)

Where

Fin is the 20 MHz input reference frequency,

M is the frequency multiplier (between 2 and 511),
N is the post divider (2, 4, 8 or 16).

The value of the internal frequency, **(Fin/8) * M**, must be between 400 and 800 MHz. In Table 1 the recommended values are shown. The values of N are set internally in the frequency synthesizer chip by the program.

Table 1. Input parameters for frequency synthesis.

F, MHz	N(formula)	N(load)	M	F step, MHz
25 < F < 50	16	3	16*F*(8/20)	0.156
50 < F < 100	8	2	8*F*(8/20)	0.313
100 < F < 200	4	1	4*F*(8/20)	0.625

Histogramming Module

The purpose of this module is to provide the logic for on-board histogramming of the data coming from a threshold scan. The module is able to interpret the data format coming from the ABCD on the datalink/LED line, extract the numbers of channels having hits from physics data packets, and increment the number of hits for these channels in the histogramming memory on the VME board. It can also clean up (write zeros to) the memory or read the number of hits for a channel and pass this information to the VME interface module.

The histogramming memory is treated as 256 blocks of 16x128 addresses (16 is the max number of chips on a module that this system could potentially work with), and 128 is the number of channels per chip. The *Set Base Address* command sets the internal pointer to one of the 16x128 blocks. This is the location where the module would store the data from a threshold scan. When one reads the number of hits from the memory the values for different channels are read out sequentially starting from address 0 in this block.

ABCD Commands Module

The main purpose of this module is to compose and send out the bitstream patterns used to communicate with the ABCD3T chip. It has the most relevance for making the threshold scans.

The commands used to issue control sequences to the chip are listed in Table 2 (Appendix A)

TV Module

This module is responsible for the test vector functionality in the system. It can load the test vector content into a dedicated memory on the VME board, send it out, receive the data from the chip, compare with the expected data, and provide the result of the comparison (weather the bitstreams matched).

A Test Vector can be thought of as a sequence of bits which define the state of the control lines going to the chip being tested on consecutive clock cycles. The Simulation Vector defines the expected chip response to the corresponding Test Vector.

The test and simulation vectors are usually stored as a sequence of bits in an ASCII file and loaded into the corresponding memory on the VME board.

The maximum TV size is 256 K clock cycles.

DAC/ADC Module

The purpose of this module is to measure some signals using an 8-channel, 12-bit ADC placed on the connector board, and to set certain parameters using many DACs in the system.

The ADC channels are listed in Table 2.

The first two channels of the ADC measure the voltages applied to the analog and digital power pins of the chip. The third channel measures temperature on the connector board. It is provided as feedback about the measurement environment. The main idea is to avoid damage to the electronics in case the cooling fan stops working. The next three channels measure internal voltages of the ABCD3T via probe pads. The last two channels provide the values of analog and

digital currents consumed by the chip by measuring the voltage across 5Ω Resistor placed in series with the chip.
The signals measured are amplified before the ADC.
DACs are set by a VME write instruction.
Functionally, they set the following quantities in the system:

1) the levels of the input signals to the ABCD, provided by pin drivers on the connector board,
2) the window comparators thresholds,
3) the timing delays.

Table 2. ADC channels. Quantity (Measurement*Coeff) gives the physical value in mV.

ADC Number	**Abbreviation**	**Coeff**	**R, Ω**	**Tcoeff,** mV/deg. C	Final quantitiy , Units
0	Vcc	**1.5**			Meas*Coeff , mV
1	Vdd	1.5			Meas*Coeff , mV
2	Temperature	0.5		10	Meas*Coeff/Tcoeff, deg. C
3	Ipreamp	0.1	250		Meas*Coeff/R , μA
4	Ishaper	0.1	1000		Meas*Coeff/R , μA
5	Vthreshold	0.5			Meas*Coeff , mV
6	Icc	0.2	5		Meas*Coeff/R , mA
7	Idd	0.2	5		Meas*Coeff/R , mA

2.4. Input/Output Signals

Pin driver chips are used (they are present in one of the two intermediate boards) to supply the input differential signals to the ABCD3T.
Window comparators, also present in the intermediate board, check the levels of the IC's output. Each (differential) output signal passes through a differential amplifier (and thus becomes single-ended) before arriving to a comparator (Fig. 6). The amplitude of the swing increases by the factor of **3**, due to the amplifier. The values of the resistors defining the amplification factor, have the nominal accuracy of **1%**. The "average" value for each signal is defined by the reference voltage applied to the amplifier. For the datalink/LED signal it is **0.29*Vdd**, for data/tokenout signals these are **0.5*Vdd**.
The uncertainty of the coefficients is also defined by the accuracy of the resistors in the network.

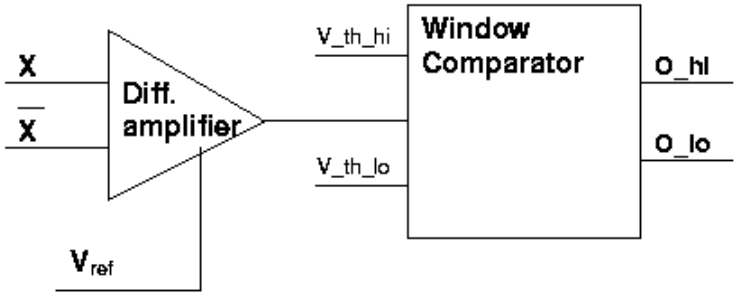

Figure 6. *ABCD3T differential output signals and the window comparator*

Two comparators are provided for each signal. One of them has the positive output if the signal level is higher than its ("high") threshold. The output of the other one is high if the signal level is below its ("low") threshold. Inside the FPGA, both the output of the first comparator and the negated output of the second comparators are checked against the expectations when running the test vectors. The signal will be valid if both data streams match the expectations. This scheme

is illustrates in Figure 7. For clarity, the signal distortion is greatly exaggerated in this diagram. The signal used in the comparison is inverted.

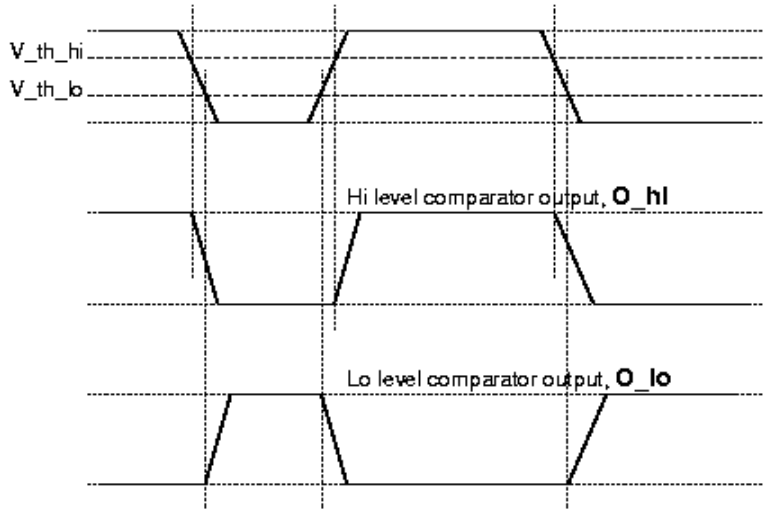

Figure 7. *Level comparators scheme*

When running the test vectors, the ID lines are being controlled by the test vector content, as indicated in Table 3.
When executing any of the commands in the Table 2 (Appendix A), for instance, when running a threshold scan, all ID lines are set to zero. Therefore, the chip address value has to be 32 (20 hex) for those commands. It is not zero due to the existence of internal bit 5, which is pulled up to high value.

3. ABCD3T Functional Tests

3.1. Analogue Tests

The goal of these tests is to determine the basic analogue parameters of the front-end: gain, noise and discriminator offset for each electronic channel. The tests (threshold scan) consist of a scan of the discriminator threshold for a given input charge applied from the internal calibration circuitry. For each threshold value a series of pulses is applied and the fraction of pulses which fire the comparator is measured. Threshold scans for different input charges are done for each channel in the chip. The threshold scan for a given input signal gives the so-called S-curve, i.e. counting rate as a fraction of triggers at the discriminator output as a function of the threshold. Provided the noise has a Gaussian amplitude distribution, the S-curve is described by the complementary error function. The 50% points of the S-curve correspond to a threshold equal to the signal amplitude while the width of the S-curve contains information about the r.m.s. value of noise (see section 2.1).
The analogue tests are performed with the FPGA Histogramming Module, using the commands in Table 2 of Appendix A to change the ABCD3T state.

The module interprets the data coming from the ABCD3T on the datalink/LED line:
1) extracts a channel number from the bitstream,
2) reads the number of hits for this channel from the histogramming memory,
3) increments the number of hits,
4) writes the new value back to the memory location (see section 2.3.1).

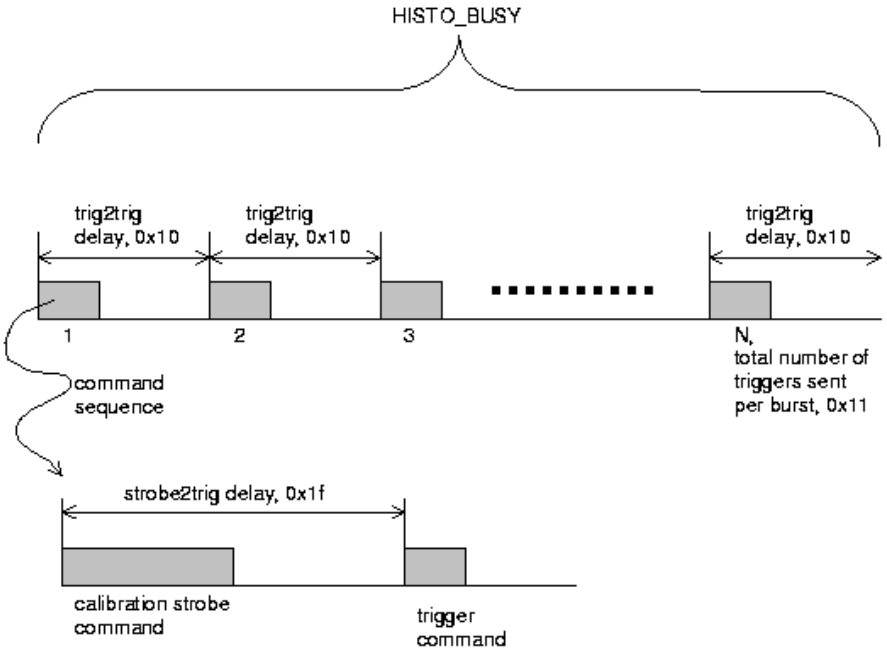

Figure 8. *Diagram of the commands in the FPGA ABCD module*

For the purposes of the threshold scan (the sequence of how to program a scan is described in Appendix B), the FPGA ABCD command module is used for issuing multiple triggers ("burst"), interspersed with calibration strobe commands (Fig.8). The user-tunable parameters are:

- *the number of triggers sent,*
  The command is write to 0x11. The corresponding register has 16 bits.
- *the spacing between the beginnings of two identical command sequences, in clock cycles,*
  The command is write to 0x10. The corresponding register has 16 bits. Note that this time includes the length of the commands bitstream.
- *the delay between the beginning of the calibration strobe command and the beginning of the trigger command, in clock cycles,*
  The command is write to 0x1f. The corresponding register has 8 bits. Note that this time includes the length of the calibration strobe command.
- *whether the calibration strobe command is present in the sequence.*
  The command is write to 0x1b. The calibration strobe command will be present in the command sequence if the bit 0 is set to 1.

3.2. Digital Tests

Several tests are applied to check all the digital functionality of the chip, mainly to verify the features of the chip related to chip control, inter-chip communication and data compression.
To determine the speed margins and the radiation resistance of the device, the digital tests are performed for different values of frequency and power supply. A chip passes a test if the output data match the simulated response.
The digital tests are performed using the FPGA Test Vector (TV) module (see Section 2.3.1).
The correspondence between the bits of the test vector and the control lines going to the chip is given in Table 3. The bits 14 through 16 are not used; they remain only for historical reasons.

The Simulation Vector defines the expected chip response to the corresponding Test Vector. The data are presented on the lower 4 bytes, of which bits 5..0 define the response per se and bits 12..8 act as a mask for them:

Data = xxxM MMMM xxxD DDDD

where D's are data lines and M's are mask lines.

The bits are defined in Table 4. As an example, datalink/LED line is used in the comparison if bit 8 is equal to one and not used otherwise.
The test and simulation vectors are usually stored as a sequence of numbers in an ASCII file.
To load the test or simulation vector into corresponding memory on the VME board, one can perform the following actions:

- reset TV/SIM memory (write to 0x01 or 0x02),
- sequentially write to TV/SIM memory the numbers comprising a vector (writes to 0x07 or 0x08),
- write to memory the value of 0x20000, signifying the end of the vector for the FPGA logic (writes to 0x07 or 0x08),
- reset TV/SIM memory (write to 0x01 or 0x02).

To exercise the test vector functionality, one needs to take the following sequence of actions:

- *load the test vector into TV memory on the VME board and simulation vector into SIM memory,*
- *set the frequency and frequency-dependent delays,*
- *reset both the Output and Resync FIFOs (write to 0x22)*
- *send the TV to the ABCD chip via Send TV command (write to 0x06),*
- *wait until the TV finishes running (TV_BUSY flags in the FPGA SR clears) and get the value of the difference in the same register. If a difference is found then the TV failed.*

The first two actions in this list can be swapped.

Table 3. Test Vector bits definition.

Bit	Control Line
0	**com0**
1	com1
2	tokenin0
3	tokenin1
4	datain0
5	datain1
6	resetB
7	masterB
8	select
9	id0
10	id1
11	id2
12	id3
13	id4
14	power_on_rst
15	test_clk
16	test_rstB

Table 4. Simulation Vector bits definition.

Bit	Control Line
0	datalink/LED
1	dataout0
2	dataout1
3	tokenout0
4	tokenout1

The TV and SIM vectors are resident on the VME board and the same TV can be run multiple times without reloading it.

To achieve high speed the TV module uses the VME board FIFOs as fast buffers. The Send TV command triggers the following sequence of actions on the VME board:

1) test vector is loaded in the output FIFO from the TV memory,
2) the FIFO is opened for reading and the TV starts to get sent out,
3) concurrently with (2) the Resync FIFO is opened for writing and it starts to accept the incoming data from ABCD,
4) when the entire TV is sent out, the Resync FIFO is closed for writing,
5) TV comparison algorithm starts to read the Resync FIFO data looking for the valid header sequence ("011101") in the datalink/LED line bitstream,
6) as soon as the header is found, the algorithm starts reading the data from the SIM memory and comparing them with the Resync FIFO data (the TV difference bit is asserted if the header has not been found by the end
   of the FIFO data stream),
7) the data comparison is finished if either of these three events occur:
   - a difference between the two data streams has been found,
   - the Resync FIFO data stream is finished,
   - the entire simulation vector have been read out from the SIM memory.

Table 5. contains the current list of test vectors, developed at CERN (courtesy of F.Anghinolfi). Also shown are stimulated I/O lines. Each TV has the resetB input line de-asserted soon after the beginning (not indicated in this Table).

Table 5. Test Vectors description.

TV	**Stimulated Input Lines**	**Affected Output Lines**	**Purpose/Description**
w1test	com0	datalink	Configuration register Write/Read test. Bits 0 thru 10 are scanned.
w2test	com0, id0, id1, id2, id3, id4	datalink	BC counter test (all 8 bits are checked). ID address bits test. Overflow function and error code test.
w3test	com0	datalink	Data Compression Logic tests with random channel mask. Having "one"s in different bits of the 3-bit hit discription.
w4test	com0,	datalink	Dynamic digital pipeline test. 4 masks. Accumulate function.
w5	com0, tokenin0, tokenin1, datain0, datain1	datalink, tokenout0, tokenout1, dataout0, dataout1	Data/to\ken bypassing circuitry test.
w6test	com0	datalink	Static digital pipeline test. 4 masks. Accumulate function test.
w6testcom1	com1, select (const)	datalink	Static digital pipeline test. 4 masks. Accumulate function.

3.3. Power Consumption Test

The FPGA DAC/ADC module is used to measure the currents and infer the power consumption on the chip (P = I V). When the current (either the digital Idd or analog Icc) is measured, the power can be obtained by multiplying the value by the voltage. Under usual running conditions, the analog voltage is Vcc = 3.5 V and the digital voltage Vdd = 4.0 V.

3.4. Test of I/O Signals Properties

The chip signal levels and phases are tested by running test vectors utilizing these signals and varying the conditions. The working range is exceeded when the test vector efficiency is below 100%.
The test system uses pin drivers chips with selectable voltage levels to supply the input differential signals to the chip. To test an input signal level, we scan the voltage swing while keeping other signals at the nominal conditions.
The digital output signals pass through window comparators only if the high voltage level is above the high threshold and the low voltage level is below the low threshold of the window comparator. To characterize the high and low voltage levels, both window comparator thresholds are set to the same value, which is varied in the scan.
Each of the input signals passes through a delay chip. The delay is scanned for a given signal while keeping other signals at nominal conditions to quantify the IC's performance with respect to input signals, phases and duty cycles.
The output data passing through window comparators go to a register chip. The delay for the clock latching data to the register is scanned to find the output signal phases.
Details of these tests can be found in [6].

4. Error Identification

Since the ABCD3T raw data stream is not read out by this system but interpreted to produce histogram data, one has to identify and provide information about the case of erroneous chip response.

4.1. Analogue Datastream Decoding

The state machine (SM) for interpreting the data coming to the FPGA during a threshold scan is shown in Figure 9 in a simplified form. The purpose of the algorithm is to extract the channel numbers containing hits from the raw data packets (DP). The SM can also recognize the *configuration* and *error* data format, although they are not expected to appear during the scan. In case when the data do not match the considered cases of
- *physics DP,*
- *no hit DP,*
- *configuration DP* or
- *error DP*

the SM goes into wait_for_trailer state, where it stays until the end of the data. It should not go there in case of correct (unscrambled) data stream.

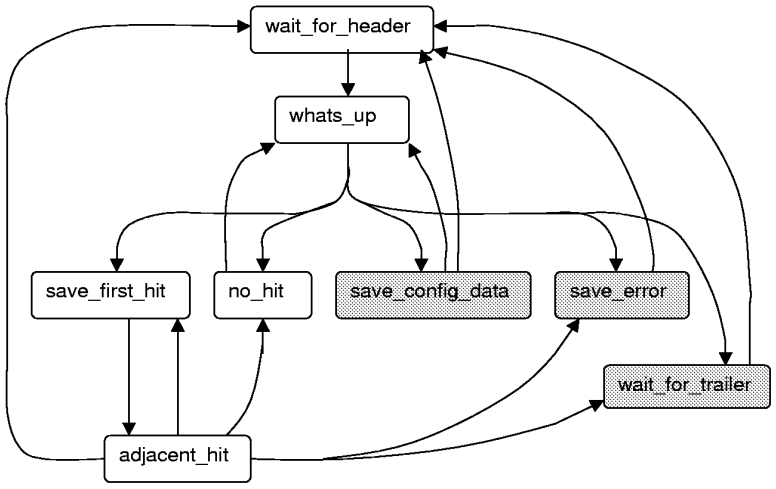

Figure 9. *Diagram of State Machine for interpreting the data during threshold scan*

Since only *physics DPs* are expected during a scan, we provide a register (read from 0x26), which has bits indicating if the SM has gone through
- *wait_for_trailer* state (bit 0), or has seen
- *error DP* (bit 1) or
- *configuration DP* (bit 2),

indicated by the gray boxes on the diagram. An asserted bit value is "1". The bits are cleared (initialized to the default zeros) by the *Send Triggers* command. The values are available for the readout after the histogramming process is done and the corresponding BUSY bits are cleared in the FPGA status register

4.2. Failure in TV Response

In the case of a TV failure the following feedback information is provided:
- identification of the location of the difference between simulation and data
- identification of the ABCD3T output signals that show the difference

5. Implementation.

The software to control the testing[7] runs on a PC under Microsoft Windows (W95/NT/2000). The PC communicates with the VME crate using the NI-VXI interface from National Instruments. The system is controlled by a Windows GUI written in Visual C++. The information is stored in data files and analyzed with the ROOT[8] framework to determine the yield.
The time needed to test one wafer containing 256 ABCD3T chips is about 6 hours[9]. A recent study on the optimization of the testing parameters, in particular for the analogue test, can be found in [10]. Sample precision calculations can be found in Appendix C.
The ASIC wafer test system will be used at the following three ATLAS Semiconductor Tracker institutions: Univ. of California at Santa Cruz, CERN and Rutherford Appleton Laboratory. The on-line control software has been integrated with the three different implementations of the probe station control code for automated wafer probing.

6. References

[1] ABCD3T ASIC Specification,
http://chipinfo.web.cern.ch/chipinfo/docs/abcd3t_spec_v1.2.pdf
[2] http://www-atlas.lbl.gov/strips/tester
[3] http://www-atlas.lbl.gov/strips/tester/VME
[4] http://www-atlas.lbl.gov/strips/tester/pin_driver
[5] http://www-atlas.lbl.gov/strips/tester/probe-card
[6] Testing I/O Signal Levels, Phases and Duty Cycles with LAST,
http://www-atlas.lbl.gov/strips/tester/doc/io_tests.pdf
[7] ABCD wafer screening, http://ific.uv.es/~lacasta/WaferTester/on-line/doc
[8] Root: An Object Oriented Data Analysis Framework, http://root.cern.ch/
[9] Testing Times, http://www-atlas.lbl.gov/strips/tester/doc/testing_times.pdf
[10] Optimization of Repetition Parameters for Threshold Scan,
http://www-atlas.lbl.gov/strips/tester/doc/opt_param_thr.pdf

7. Appendix A. VME Board Commands and Addressing

The following is the list of the bits used in the VME address (32 bits):

Bits 31..25 = 8-bit VME board address = set by dip switches on the board
Bits 24..10 = not used
Bits 9..4 => to execute one of the VME commands listed in Tables 1,2,3
Bits 3..0 = not used

The VME commands used for board operations and their description are listed in Table 1. The VME commands to issue all control sequences to the chip are listed in Table 2. Details on the sequence, can be found in the ABCD specifications document.

Table 1. VME Commands for Board Operation.

Bits 9..4 hex	Read/Write	Description	VME Data Bus
00	Read	Read FPGA Status Register	Bits 11..0 = ACE Bit 12 = HISTO_1_BUSY Bit 13 = HISTO_2_BUSY Bit 14 = DAC_ADC_BUSY Bit 15 = TV_BUSY Bit 16 = BUSY (4 BUSY bits above OR'd together) Bit 17=TVDifferenceFound Bits 31..18 = 0
01	Write	Reset Test Vector Memory Counter to 0	X
02	Write	Reset Simulation Vector Memory Cnt to 0	X
03	Write	Clear Histogram Memory	X
04	Read	Read from Histogram Memory and Increment Memory Pointer	Bits 15..0 = Stream B Bits 31..16 = Stream A
05	Write	Set Base Address for Histogramming and Reset Histogram Memory Counter to xaa000000, where "aa" stands for the Base Address	Bits 7..0 = Base Address Bits 31..8 = X
06	Write	Send Test Vector	
07	Write	Write to Test Vector Memory and Increment the Test Vector Memory Counter	Bits 17..0 = data Bits 31..18 = X
08	Write	Write to Simuation Vector Memory and Increment the Simulation Vector Memory Counter	Bits 17..0 = data Bits 31..18 = X
09	Write	Reset the ReSync FIFO read pointer	X
0a	Write	Set DAC	Bits 9..0 = data Bits 22..19 = dac address Bits 18..16 = dac channel other bits: X
0b	Write	Send Convert Signal to ADC	Bits 2..0 = ADC number
0c	Read	Read ADC Data from last Conversion	Bits 11..0 = data
0d	Read	Read from ReSync FIFO	Bits 8..0 = data Bits 17 = FIFO empty flag: 0 = empty, 1 = not empty Bits 16..9, 31..18 = X
0e	Write	Start sending triggers and decode and histogram the data	X
0f	Write	Set Frequency	Bits 8..0 = M Bits 10..9 = N Bits 13..11 = T

Table 2. VME Commands used to issue control sequences to the chip.

Bits 9..4 hex	Read/Write	Description	VME Data Bus
10	Write	Set Trigger-To-Trigger Delay	Bits 15..0 = delay in cc (25ns) Bits 31..16 = X
11	Write	Set Number of Triggers to be sent per Burst	Bits 15..0 = #triggers Bits 31..16 = X
12	Write	Send Soft Reset (all chips)	X
13	Write	Send BC Reset (all chips)	X
14	Write	Write to Configuration Register	Bits 21..16 = chip addr. Bits 15..0 = config reg data Bits 31..22 = X
15	Write	Reset FPGA-Mask Register Pointer	X
16	Write	Write 32 bits to FPGA-Mask Register	31..0 = mask reg data
17	Write	Send FPGA-Mask Register to a chip	Bits 21..16 = chip addr. Bits 15..0 = X Bits 31..16 = X
18	Write	Load Strobe Delay Register	Bits 15..0 = reg data Bits 21..16 = chip addr. Bits 31..22 = X
19	Write	Load Threshold/Cal DAC Ampl Reg	Bits 15..0 = reg data Bits 21..16 = chip addr. Bits 31..22 = X
1a	Write	Enable Data Taking	Bits 21..16 = chip addr. Bits 15..0 = X Bits 31..16 = X
1b	Write	Select/Strobe Enable	Bits 1: abcd select bit Bits 0: strobe enable bit
1c	Write	Issue Hard Reset	not implemented
1d	Write	Load Bias DAC	Bits 15..0 = reg data Bits 21..16 = chip addr. Bits 31..22 = X
1e	Write	Load Trim DAC	Bits 15..0 = reg data Bits 21..16 = chip addr. Bits 31..22 = X
1f	Write	Set Strobe-To-Trigger Delay	Bits 7..0 = delay in cc Bits 31..8 = X

Table 3. Additional VME Commands.

Bits 9..4 hex	Read/Write	Description	VME Data Bus
20	Read	TV Difference location in time sequence	Bits 17..0 = Data (=0 if no difference) Bits 31..18 = 0
21	Read	TV Difference Word	Bits 7..0 = difference seen on the corresp. TV output lines at the 1st difference location; (bit #N = 1 if there was a difference on line #N; =0 otherwise) Bits 31..8 = 0
22	Write	Reset both Output and Resync FIFOs	X
23	N/A	Nothing is implemented here	N/A
24	Write	Tristate ABCD input signals	Bit 0 is data, the input signals are valid if 0, tristated if 1.
25	Write	Select hit pattern to decode the data for the histogramming	Bits 2..0 (see Table 4)
26	Read	Histogramming data decoding (error) flags	Bits 2..0
27	Read	Read the firmware version date.	Bits 2..0 = Year Bits 6..3 = Month Bits 11..7 = Day

Table 4. Hit pattern selection.

Value	Pattern	Name
0 (default)	1XX or X1X or XX1	Hit mode
1	X1X	Level mode
2	01X	Edge mode
3	XXX	Test mode
4	111	All hits
5	100	1st hit only
6	010	2nd hit only
7	001	3rd hit only

8. Appendix B. How to Program a Scan

A sample sequence of actions issued for a threshold scan is listed below.
Before issuing any command, one has to check with the “read status register” command (x00) that the busy flag (bit 16) is not set.

1) Initialize Module
   - Set Frequency to 40 MHz (x0F)
   - Set DACs for the power, pin drivers voltage and window comparators
   - Load Mask Registers
   - Load Trim DAC’s of all channels (x1E).
   - Load Strobe Delay Register (x18)
   - Load Bias DAC’s (x1D)
   - Load the ABCD Configuration Register
2) Initialize Scan Parameters
   - Clear Histogram Memory (x03)
   - Set Number of Triggers per scan step (“burst”) (x11)
   - Set Trigger to Trigger delay (x10). Make sure you allow enough readout time.
   - Set Strobe to Trigger delay (x1F). This should be around 129.
   - Enable Calibration Strobe.
3) Do the Scan
   Loop over the following 5 commands (i = [0..number of scan points-1])
   - Enable Data Taking
   - Set the scan parameter (eg. threshold) to the i-th scan point (eg. x19)
   - Set Base Address to the actual step number i (x05)
   - Send burst of triggers (x0E)
   - Wait while the Histogramming Module is busy
4) Read Histogram Memory
   Loop over the following sequence for each scan point:
   - Set base address to the value for this point (x05)
   - Read from Histogram Memory (x04) the number of channels you want (128 times if all channels are enabled).

9. Appendix C. Sample Precision Calculations from the Optimization of Threshold Parameters Study

Here are some sample calculations. The values used for the input parameters are currently the default running values for these parameters.

Noise (σ) ~ mV	5
Voltage Step (ΔV) ~ mV	7.5
n (# of scan points)	4
Gain ~ mV/fC	50
Offset ~ mV	15
Trim Slope	3
Nevts	500

Scan points ~ fC
2.5
3.0
3.5
4.0

The percent statistical errors for various numbers of trigger events are in the tables below.

***ΔV* = 7.5mV (default):**

Nevts	%δ(Gain)	%δ(Offset)	%δ(Noise)	%δ(Trim Slope)
100	0.823	9.046	12.309	1.345
200	0.582	6.396	8.682	0.951
300	0.475	5.223	7.083	0.776
400	0.411	4.523	6.131	0.672
500	0.368	4.045	5.483	0.601
600	0.336	3.693	5.004	0.549
700	0.311	3.419	4.632	0.508

***ΔV* = 5.0mV:**

Nevts	%δ(Gain)	%δ(Offset)	%δ(Noise)	%δ(Trim Slope)
100	0.672	5.539	10.050	1.098
200	0.475	3.917	7.089	0.776
300	0.388	3.198	5.783	0.634
400	0.336	2.770	5.006	0.549
500	0.300	2.477	4.477	0.491
600	0.274	2.261	4.086	0.448
700	0.254	2.094	3.782	0.415

***ΔV* = 5.0mV:**

Nevts	%δ(Gain)	%δ(Offset)	%δ(Noise)	%δ(Trim Slope)
100	0.475	3.917	7.107	0.776
200	0.336	2.770	5.013	0.549
300	0.274	2.261	4.089	0.448
400	0.238	1.958	3.540	0.388
500	0.212	1.752	3.165	0.347
600	0.194	1.599	2.889	0.317
700	0.180	1.480	2.675	0.293

The percent errors in the tables above are given as characteristic only, as they rely upon values of the respective parameters as given in the table above.